Manual and Automatic Identification of Similar Arguments in EFL Learner Essays

Ahmed Mousa[1], Ronja Laarmann-Quante[2] and Andrea Horbach[3,4]
[1]University of Duisburg-Essen, Germany,
[2]Ruhr University Bochum, Faculty of Philology, Department of Linguistics, Germany,
[3]CATALPA, FernUniversität in Hagen, Germany,
[4]Universität Hildesheim, Germany

Abstract

Argument mining typically focuses on identifying argumentative units such as *claim, position, evidence* etc. in texts. In an educational setting, e.g. when teachers grade students' essays, they may in addition benefit from information about the content of the arguments being used. We thus present a pilot study on the identification of similar arguments in a set of essays written by English-as-a-foreign-language (EFL) students. In a manual annotation study, we show that human annotators are able to assign sentences to a set of 26 reference arguments with a rather high agreement of $\kappa > .70$. In a set of experiments based on (a) unsupervised clustering and (b) supervised machine learning, we find that both approaches perform rather poorly on this task, but can be moderately improved by using a set of six meta classes instead of the more fine-grained argument distinction.

1 Introduction

Argumentative essays are frequently written as part of foreign language instruction. A common natural language processing (NLP) task on these kinds of texts is argument mining, the task of automatically detecting argumentative units in texts (Lawrence and Reed, 2020). In argument mining, arguments are typically categorized according to their function, such as *claim, position, evidence* etc., but most argument mining approaches do not offer methods to categorize the content covered by a particular argument.

From an educational perspective, however, knowing which sub-topics of a certain prompt are addressed where in the essay could be beneficial both for summative and formative feedback. For example, while grading an essay, teachers could benefit from knowing how many different arguments or how many pro and con arguments occur and how they are distributed in the text. The automatic identification of arguments also allows for an easier comparison of the content of different essays. Students could receive such information as feedback. Figure 1 shows an example of an argumentative essay and how the information could be highlighted in the text.

This paper presents a pilot study on the automatic identification of similar arguments in texts of EFL students. We want to find out (a) how well human annotators agree when detecting similar arguments and (b) what performance on this task can be achieved with an automatic model and whether a supervised approached with limited training data or an unsupervised clustering approach works better. To do so, we conduct an annotation study in which we first determine a set of reference arguments found in the essays. By 'reference argument' we mean a statement that summarizes in one sentence the core of an argument found in one or more essays.

We then use these reference arguments to annotate a subset of the dataset for computing inter-annotator agreement and to be used as gold standard for evaluating automatic models. In our experiments, we compare variants of k-means clustering using different seed sets and vectorization methods. We evaluate them according to their ability to place gold segments with the same cluster ID in the same cluster and unrelated segments in different clusters and compare them with a supervised Machine Learning (ML) approach. We either distinguish between fine-grained arguments or merge different arguments into meta-classes such as *Pro*, *Contra* or *Irrelevant*.

Thus, our paper contributes to the research on similar argument identification in two ways. Firstly, we provide manual annotations of similar arguments for a set of EFL learner texts. We

Ahmed Mousa, Ronja Laarmann-Quante and Andrea Horbach. Manual and Automatic Identification of Similar Arguments in EFL Learner Essays. *Proceedings of the 12th Workshop on Natural Language Processing for Computer Assisted Language Learning (NLP4CALL 2023)*. Linköping Electronic Conference Proceedings 197: 85–93.

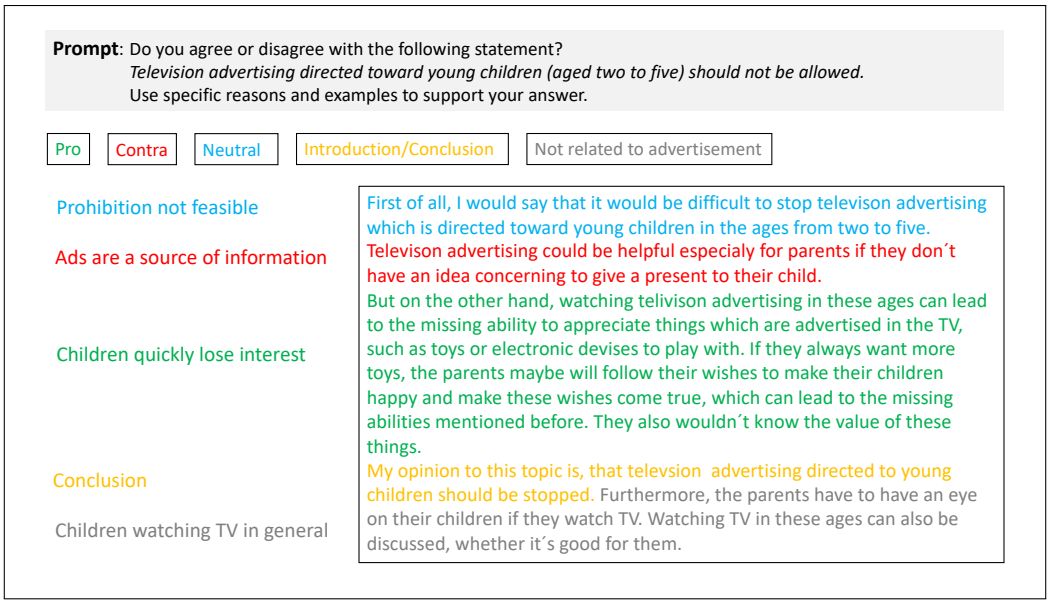

Figure 1: Example of an essay annotated with argumentative units and argument summaries.

make our annotated dataset available under https://github.com/andreahorbach/ArgumentClustering. Secondly, we provide a number of baselines results for automating the task based on different methods and for different levels of granularity.

2 Related Work

2.1 Argument Identification

Argument mining usually deals with identifying certain argument types based on their function in the text (Wachsmuth et al., 2016; Nguyen and Litman, 2018; Ding et al., 2022). While most such approaches work in a supervised way, Persing and Ng (2020) use an unsupervised approach to bootstrap argumentative units of different types based on a seed set obtained from applying simple heuristics. Our approach is related to argument mining but has the major difference that the goal is to classify any identified segment based on its content. In educational contexts, even when scoring argumentative essays, argumentative content is rarely explicitly focused on. In datasets such as the ASAP essay dataset[1] argumentative essays are either scored holistically or according to categories such as overall content, organization fluency etc. The content of individual arguments, however, is only rarely explicitly addressed. Horbach et al. (2017), for example, conduct experiments on German essays based on an annotation scheme indicating the presence or absence of certain arguments regarding a topic, but they do not mark the exact location in the text.

2.2 Text Clustering

In the educational domain, clustering techniques have been employed to support automatic scoring of learner answers with the basic idea that answers appearing in the same cluster likely convey the same content and can therefore be graded together. Proposed approaches rely on surface representations (Horbach et al., 2014), semantic representation such as LSA (Zehner et al., 2016) or a combination thereof (Basu et al., 2013).

In essay scoring, clustering techniques have been used on the text level, such as Chen et al. (2010), who clustered an essay corpus into the number of different scores found in the data. On a more fine-grained level, and probably the closest to our study, Chang et al. (2021) annotate and cluster sentences in Finnish student essays based on their argumentative content. Besides clustering, they use an information retrieval approach but no supervised machine learning like we do.

3 Dataset and Manual Annotations

3.1 MEWS Dataset

We conduct our experiments on the MEWS dataset (Measuring Writing at Secondary Level; Keller, 2016). It consists of English essays written by 10th grade students in Germany and Switzerland who learn English as a foreign language. The

[1] https://www.kaggle.com/competitions/asap-aes/overview

Method	# segments	Avg. # tokens
Sentences	38,715	18.79
W/ Connectives	37,505	19.27

Table 1: Average number and length of segments per essay for each segmentation method.

dataset contains four individual writing prompts, two for independent and two for integrated essays. In this paper, we focus on one of the two independent argumentative writing prompts, in which the learners are supposed to state whether they agree or disagree with a statement and to provide reasons for their answer. The prompt is: *Television advertising directed toward young children (age 2 to 5) should not be allowed*. In total, the dataset contains 2,382 essays in response to this prompt.

3.2 Argumentative Units

We consider different options to automatically segment the essays into units that can be clustered or labeled as different arguments. First, we looked into splitting at paragraph boundaries but as many learners did not arrange their texts into multiple paragraphs this approach turned out to be not feasible. Second, we consider **sentences**, which are an obvious linguistic unit and easy to extract. The potential shortcoming is that a sentence may contain more than one argument or an argument may stretch over multiple sentences. As an alternative, we split the texts using a comprehensive list of 215 **discourse connectives** such as *furthermore*, *on the other hand*, *in conclusion* as separators. In this segmentation variant, we only split at sentence boundaries when the next sentence starts with such a connective to indicate that a new argument is following. We decided not to split at discourse connectives within a sentence because we found that it too often leads to uninterpretable text snippets.

Table 1 shows the average number and length of segments found by either variant. We see that the two variants do not differ much numerically from each other. Upon manual inspection, we found that they indeed produced very similar results. Part of the reason may be that the learners do not use discourse connectives consistently. For the sake of simplicity, we therefore decided to use sentences as units, although in future work a proper argumentative unit detection based on gold standard segmentation might be a better alternative.

3.3 Annotation of Gold Standard Arguments

To create a gold standard, we used a two-step process.

Step 1: Determining the Number of Reference Arguments First, we determined how many different arguments there are in the dataset. To do this in a time-efficient manner, one annotator looked at a number of essays and compiled a list of found arguments and the corresponding sentences in an iterative process until no new arguments were detected in four subsequent essays. This happened after a total of 14 essays. There were no specific guidelines for this step. Then, a second annotator looked at the same set of essays and independently collected all different arguments that he found, i.e. he did not see which arguments annotator 1 had collected before. Together with two additional adjudicators, a final set of 26 **reference arguments** was compiled. Each reference argument consists of a short summary of the core content of the argument (produced by the annotators) and a set of sentences from the essays that correspond to this argument. See Table 2 for some examples.

There are some 'special' types of reference arguments worth mentioning: *Introduction* and *Conclusion* refer to all introductory or concluding sentences of an essay, which do not contain arguments per se, *Non-English* refers to all sentences written in a different language (e.g. when students copied material from the German instructions) and *Irrelevant*, which refers to sentences that are meta-comments or do not refer to the prompt e.g. *Sorry for not writing anything*. Furthermore, we added one additional category called *New Arguments* to account for arguments not detected before.

Step 2: Annotating Arguments in Text In the next step, the same two annotators were given the list of reference arguments that were compiled in step 1 and annotated a set of 235 sentences from new essays with the reference arguments they correspond to. We aimed at a set of sentences that would cover all reference arguments. To approximate this, we automatically clustered all sentences from the essays as described in Section 4.1 (with the reference arguments as centroids and tf-idf vectorization) and picked five random sentences from each cluster for the manual annotation. The annotators agreed in 169 out of 235 annotated sentences, reaching an inter-annotator

Argument summary	Corresponding sentences from the essays
Advertisements can have positive effects on children's behavior.	Advertisement for children does not have to be a bad thing, it can be used to influence them so that their behaviour will have a positive effekt on society and nature.
	But that argument is quite small since the children might want something for the outdoor fun like a new special ball and so they want to play outside and stop sitting in front of the TV and that can´t be bad at all.
It does not really matter because young children normally do not watch TV that often or shouldn't be allowed to.	I also remember me having fun to go outside and not having to worry about an television advertisement
	Also one has to add that young children aged two to five normally do not watch TV that often. Therefore it does not really matter
	there seems to be no need for a prohibition of especially this type of advertisements since most of the children aged 2 to 5 are allowed to watch television
Young children are easily manipulated by advertisements.	The advertisment has an influence on the Children and in this age they don't know when they are under an influence
	Chlidren form the age of two to five have not been able to develop their own charcter jet, that makes them an easy target for advertisement
	Because they are so easy to influence and probably believe the things that are said, even though they are not true.

Table 2: Examples of manually identified arguments and corresponding sentences from the essays. We refer to these as reference arguments.

agreement of Cohen's $\kappa = 0.718$. After the annotators were shown where they disagreed, one annotator corrected six obvious errors, raising the inter-annotator agreement to 0.732. This rather high agreement value shows that despite the large number of reference arguments and the overall diverse texts (resulting from an independent rather than integrated writing prompt), arguments in student essays can be clustered consistently – with the limitation that only one prompt was analyzed in this study.

The major sources of disagreements (24 and 20 cases, respectively) were that one annotator tended to assign arguments to the *New Argument* or *Irrelevant* category, respectively, while the other annotator would assign them to one of the existing reference arguments. We chose the annotations of the annotator who preferred to assign the arguments to the existing reference arguments as the final gold standard for our evaluation.

The most frequently occurring arguments/categories are *Irrelevant* (11.5%), *Children shouldn't watch TV in general* (8.1%) and *Children are easily manipulated by advertisements.* (8.1%). Two arguments were found only once, namely *Children may adopt undesired behavior from advertisements* and *Children want to be treated like adults*.

4 Argument Identification Experiments

4.1 Experimental Setup

In our experiments, we compare several instantiations of k-means clustering with supervised machine learning.

Clustering algorithm The basic k-means algorithm (Arthur and Vassilvitskii, 2006) iteratively assigns elements to be clustered to the closest instance from a set of centroids. These centroids are often randomly chosen in the first iteration, later the centroid of each cluster from the previous round is used until the cluster assignment is stable. We choose the number of clusters k to be 26, i.e. the number of reference arguments we manually identified as described in Section 3.3.

One obvious parameter in the setup of k-means clustering is the choice of a suitable **distance metric** between items operationalized by the vectorization method to be combined with cosine similarity. We use four different methods. Cosine similarity between **tf-idf weighted ngram** features is a baseline relying on surface features. We compare it with three embedding-based methods, also using cosine similarity. First we average word vectors using pretrained word embeddings from Word2Vec (Mikolov et al., 2013) or FastText (Joulin et al., 2016) to create sentence vectors. Second, we make use of Sentence-BERT (SBERT, Reimers and Gurevych, 2019) to create

an embedding vector per sentence.[2]

A second parametrization of k-means concerns the initialization of seed centroids. We either use random sentences as seeds (**random seeds**) or use our manually annotated reference arguments as centroids (**gold centroids**) by averaging over sentence vectors for all sentences identified for a reference argument as described in Section 3.3. We assume that our gold centroids are already optimal in a sense that they represent the individual arguments in the essays, therefore we stop after one round of clustering in the **gold centroids** setup. In the **random seeds** setup, we iterate as usual until the clustering is stable, i.e. until cluster assignments do not change anymore.[3]

Supervised approach As an alternative, we explore a supervised machine learning approach using logistic regression with different feature setups: tf-idf weighted n-grams or SBERT vectors. We perform 10-fold cross validation on the manually annotated gold-standard sentences from Section 3.3 with cluster ID as the target label. That means, in each iteration, we train on about 212 sentences, which is a rather small number of instances given the 26 target classes.

Evaluation Metrics As we do not have a fully annotated gold-standard cluster assignment for every sentence in the dataset, we rely on the subset of human annotations described in Section 3.3, meaning that most established cluster evaluation techniques (Amigó et al., 2009) are not applicable to our evaluation setup in a straightforward manner. Furthermore, we cannot easily say which cluster represents which reference argument (i.e. which gold-standard label) in order to report instance-based accuracy. Therefore we adapt pair-counting cluster evaluation methods (Halkidi et al., 2001) that use only the annotated subset of sentences in the clusters. From this annotated subset, we form pairs of sentences which belong either into the same cluster or into different clusters according to the gold standard. We thus evaluate for every clustering what percentage of same-cluster pairs was indeed clustered into the same cluster and how many different-cluster pairs ended up in different clusters, as well as using the established Jaccard coefficient J:

$$J = \frac{SS}{SS + SD + DS} \quad (1)$$

where SS ('same-same') is the number of pairs that belong into one cluster according to the gold standard and are assigned to the same cluster by the algorithm, SD ('same-different') is the number of pairs that are in the same gold cluster but ended up in different clusters in the algorithm and DS ('different-same') the opposite case. The Jaccard coefficient thus ranges from 0 to 1 with 1 being the best possible value. In addition, we report precision and recall, which refer to 'same'-pairs as the positive class, and overall accuracy. One has to be aware that for the pairwise evaluation, accuracy is overall high due to the high number of DD ('different-different') pairs.

4.2 Experiment 1 - Fine-Grained Argument Distinction

Comparison of Clustering Algorithms and Vectorization Methods In a first set of experiments, we compare the different vectorizing approaches for the two variants (gold centroids vs. random seeds) of k-means. The results are shown in Table 3.

We observe that, against our initial expectations, there is no clear advantage of using gold centroids over random seeds. In terms of accuracy and Jaccard, the gold centroids work slightly better than random seeds when tf-idf or FastText is used for vectorization but overall, the differences are rather small. When comparing the different vectorization methods, SBERT and Word2Vec outperform the other two methods for most evaluation metrics. The overall best clustering result is achieved with k-means with random seeds using SBERT, but only reaching a Jaccard index of .115.

We cannot directly compare the (unlabeled) clusters to the gold standard but we can compare the distribution of cluster size. For each clustering setup, we order clusters by size in descending order and plot the cluster size. A horizontal line would mean that all clusters have the same size. A steeply falling line which then becomes flat would mean that there are few clusters with many instances and many clusters with only few instances

[2] We use the following pre-trained models: https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?usp=sharing (Word2Vec), https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.bin.gz (FastText), all-mpnet-base-v2 from https://www.sbert.net/docs/pretrained_models.html (SBERT).

[3] We also tried a mix of both, i.e. starting with gold seeds and then iterating until the cluster assignments are stable. However, since the results were overall worse than for the gold centroids setup, we will not report them in detail for space reasons.

	Vectorization	SS	DD	DS	SD	Acc.	Prec.	Rec.	Jaccard
k-means	tf-idf	240	20,497	3,260	979	.830	.197	.069	.054
	SBERT	246	22,846	911	973	.925	.202	.213	.115
	Word2Vec	258	22,102	1,655	961	.895	.212	.135	.090
	FastText	202	20,793	2,964	1,017	.841	.166	.064	.048
gold centroids	tf-idf	185	22,730	1,027	1,034	.917	.152	.153	.082
	SBERT	200	22,893	864	1,019	.925	.164	.188	.096
	Word2Vec	244	22,243	1,514	975	.900	.200	.139	.089
	FastText	181	22,201	1,556	1,038	.896	.148	.104	.065
supervised ML	tf-idf	812	9,239	14,518	407	.402	.666	.053	.052
	SBERT	589	19,148	4,609	630	.790	.483	.113	.101

Table 3: Results of Experiment 1: Fine-grained argument distinction. Comparison of different clustering techniques and supervised machine learning.

	Vectorization	SS	DD	DS	SD	Acc.	Prec.	Rec.	Jaccard
k-means	tf-idf	2,803	10,571	8,410	3,192	.536	.468	.250	.195
	SBERT	1,393	16,209	2,772	4,602	.705	.233	.335	.159
	Word2Vec	2,047	12,813	6,168	3,948	.595	.342	.299	.168
	FastText	2,108	12,993	5,988	3,887	.605	.352	.260	.176
gold centroids	tf-idf	2,276	14,851	4,130	3,719	.686	.380	.355	.22
	SBERT	2,010	15,559	3,422	3,985	.703	.335	.370	.21
	Word2Vec	2,267	14,237	4,744	3,728	.661	.378	.323	.21
	FastText	2,302	14,339	4,642	3,693	.666	.384	.332	.22
supervised ML	tf-idf	3,311	10,065	8,916	2,684	.536	.552	. 271	.222
	SBERT	3,241	12,489	6,492	2,754	.630	.541	.333	.260

Table 4: Results of Experiment 2: Distinction of broader argument classes: Comparison of different clustering techniques and supervised machine learning.

(like a zipf curve). Figure 2 shows the results for the random seeds setup in comparison with the gold standard. We see that in the gold standard (solid red line), most clusters have roughly the same size. For clusters with tf-idf and FastText vectorization, however, we see that there are a few very dominating clusters with many instances. Overall, the SBERT curve looks most similar to the gold standard.

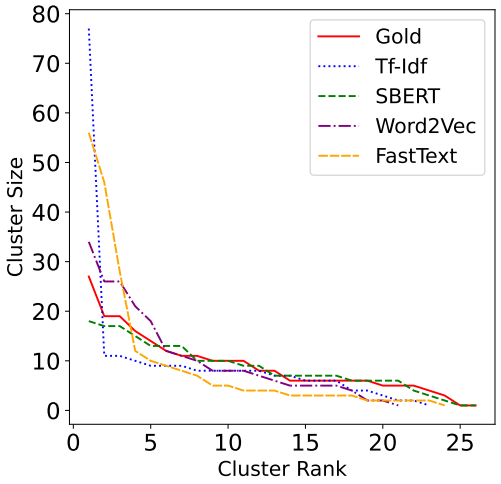

Figure 2: Cluster sizes of the gold standard clusters and the clusters produced by k-means with random seeds and different vectorization methods.

Comparison with Supervised ML The results of the supervised ML experiments based on pairwise evaluation is shown in the lower part of Table 3. As in the unsupervised clustering setup, we see that SBERT features outperform tf-idf based features in terms of accuracy and Jaccard index. Overall, with a maximum Jaccard index of .10, the performance of the supervised ML approach is lower than the best unsupervised clustering setup. This is probably due to the limited amount of labeled training data and the high number of classes.

When we look at the number of correctly assigned instances, we achieve a classification accuracy of .31 (SBERT) and .23 (tf-idf), respectively. What is particularly striking about the results is that SBERT assigns sentences only to 10 out of the 26 reference arguments (tf-idf: 8 out of 26). Unsurprisingly, most sentences are assigned the labels that occurred most frequently in the manually annotated training data.

Class	# Ref. Args.
Pro	9
Contra	9
Neutral	4
Irrelevant	2
Intro	1
Conclusion	1

Table 5: Distribution of reference arguments over the merged classes.

4.3 Experiment 2 - Distinction of Broader Argument Classes

In the previous experiment, we found that the results for distinguishing between individual arguments were rather unsatisfactory. Especially for the supervised ML approach, this may be due to the imbalance of a high number of classes and rather few training instances. Therefore, we conduct a second set of experiments in which we merge the 26 reference arguments into six meta-classes: *Pro*, *Contra*, *Neutral*, *Irrelevant*, *Introduction*, *Conclusion*. Table 5 shows how many reference arguments fall into which class. We see that there are as many different pro arguments as contra arguments in our set of manually identified arguments.

We repeat our experiments on these broader argument classes, i.e. setting k to 6 in the clustering experiments. The results are shown in Table 4. We see that compared to the fine-grained argument distinction, the overall accuracy drops in the pairwise evaluation setup because of the smaller number of different-different pairs. In terms of precision, recall and Jaccard index, we see that the clustering works better in the merged classes setup than in the fine-grained setup. Furthermore, the differences between the different vectorization methods are again rather small but unlike in the fine-grained setup we see a slight advantage of using gold centroids over random seeds.

The supervised machine learning approach again performs worse than the unsupervised clustering, but only in terms of accuracy. With SBERT features, the supervised ML approach reaches a Jaccard index of .26, outperforming both the tf-idf features as well as the unsupervised clustering. When looking at instance-based classification accuracy of the supervised ML approach, we get an accuracy of .46 for tf-idf based features and .53 for SBERT features. However, the overall accuracy is misleading. Figure 3 shows the distribution of

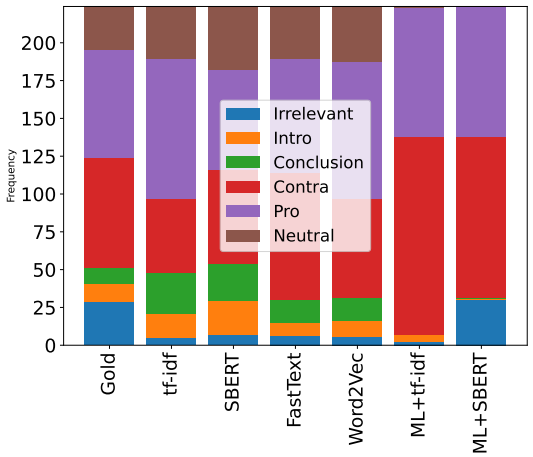

Figure 3: Distribution of argument classes in the gold standard (left), and in the outcome of the clustering and machine learning experiments.

classes in the gold standard (leftmost bar) and in the two ML setups (two rightmost bars). We see that with SBERT features, the algorithm never assigns sentences to the *Conclusion* or *Neutral* class and hardly any to *Introduction*. With tf-idf features, almost 60% of the sentences are assigned to the *Contra* class, which does not reflect the distribution in the gold standard at all.

For comparison, the four bars in the middle show the distribution resulting from the unsupervised clustering with gold centroids. We assigned the labels to the clusters by propagating the majority label of the annotated sentences to the whole cluster.[4] We see that their distributions are much closer to the gold standard but underestimate the number of *Irrelevant* arguments and overestimate the number of *Conclusion* sentences.

5 Discussion and Implications for Practice

Our experiments clearly show that fine-grained argument distinction is rather hard to perform – both with unsupervised clustering and supervised machine learning with rather limited training data (about 200 sentences – probably still more than one could expect in a natural classroom situation).

In an ideal teaching scenario, all sentences from a set of student essays would be clustered automatically, without manual annotation effort. In our study, we used k-means as clustering algorithm, and found that cluster assignment based on

[4] Such a procedure was not feasible in the fine-grained setting due to the large number of classes.

random seeds works as well as explicitly setting gold centroids, which implies that no manual intervention would be required at this step. However, for k-means it is required to set the expected number of outcome clusters. This, in turn, requires that the number of different arguments that can occur is known. Our approach from Experiment 2, i.e. merging the arguments into six broad meta-classes, would overcome this issue in that these classes do not depend on the essay topic. We found that reducing the number of classes also improves the performance. However, highlighting these classes in an essay would convey information about argumentation structure rather than about the content of the argumentation.

6 Conclusion and Outlook

We presented a pilot study for the automatic identification of similar arguments in students' EFL essays. In an annotation study, we found that human annotators are able to assign sentences to a set of reference arguments with a rather high agreement of $\kappa > .70$. Our machine learning experiments showed that for both supervised ML and unsupervised clustering the performance for distinguishing between a set of 26 different arguments was rather poor. In a second set of experiments based on broader argument classes, a better performance could be achieved at the cost of losing information about essay content. Our experiments were based on essays from a single prompt only. In future work, we want to extend both the manual annotation study as well as the ML experiments to a larger set of essays from different topics and prompts.

Acknowledgments

This work was partially conducted at "CATALPA - Center of Advanced Technology for Assisted Learning and Predictive Analytics" of the FernUniversität in Hagen, Germany.

References

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12:461–486.

David Arthur and Sergei Vassilvitskii. 2006. k-means++: The advantages of careful seeding. Technical report, Stanford.

Sumit Basu, Chuck Jacobs, and Lucy Vanderwende. 2013. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 1:391–402.

Li-Hsin Chang, Iiro Rastas, Sampo Pyysalo, and Filip Ginter. 2021. Deep learning for sentence clustering in essay grading support. *arXiv preprint arXiv:2104.11556*.

Yen-Yu Chen, Chien-Liang Liu, Tao-Hsing Chang, and Chia-Hoang Lee. 2010. An unsupervised automated essay scoring system. *IEEE Intelligent systems*, 25(05):61–67.

Yuning Ding, Marie Bexte, and Andrea Horbach. 2022. Don't drop the topic - the role of the prompt in argument identification in student writing. In *Proceedings of the 17th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2022)*, pages 124–133, Seattle, Washington. Association for Computational Linguistics.

Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. 2001. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17:107–145.

Andrea Horbach, Alexis Palmer, and Magdalena Wolska. 2014. Finding a tradeoff between accuracy and rater's workload in grading clustered short answers. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 588–595.

Andrea Horbach, Dirk Scholten-Akoun, Yuning Ding, and Torsten Zesch. 2017. Fine-grained essay scoring of a complex writing task for native speakers. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 357–366.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Stefan Keller. 2016. Measuring Writing at Secondary Level (MEWS). Eine binationale Studie. *Babylonia*, 3:46–48.

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Huy Nguyen and Diane Litman. 2018. Argument mining for improving the automated scoring of persuasive essays. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Isaac Persing and Vincent Ng. 2020. Unsupervised argumentation mining in student essays. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6795–6803, Marseille, France. European Language Resources Association.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Henning Wachsmuth, Khalid Al Khatib, and Benno Stein. 2016. Using argument mining to assess the argumentation quality of essays. In *Proceedings of COLING 2016, the 26th international conference on Computational Linguistics: Technical papers*, pages 1680–1691.

Fabian Zehner, Christine Sälzer, and Frank Goldhammer. 2016. Automatic coding of short text responses via clustering in educational assessment. *Educational and Psychological Measurement*, 76(2):280–303.